Configurer le dictionnaire ESA en utilisant l'API REST Python et AsyncOS

Table des matières

Introduction

Ce document décrit comment utiliser un script Python pour ajouter un nouveau dictionnaire à un dispositif de sécurité de la messagerie Cisco (ESA) à l'aide de l'API AsyncOS (API REST).

Conditions préalables

Vérifiez ces conditions préalables :

- langage de programmation Python
- module de `requêtes` Python
- JSON
- Appliance de sécurisation de la messagerie Cisco (ESA)

Exigences

Utilisez le logiciel et l'accès suivants pour exécuter le script :

- Appliance de sécurisation de la messagerie Cisco (ESA) exécutant AsyncOS 14.x ou version ultérieure
- Python 3

- module de `requêtes` Python
- Identifiants d'administrateur ESA avec autorisation d'utiliser l'API AsyncOS (API REST)
- Accès à l'interface de gestion ESA via HTTPS sur le port `6443` (recommandé)

Composants utilisés

Les composants suivants ont été utilisés pour effectuer ces travaux pratiques :

1) Cisco ESA - 15.0.0-104

2) Éditeur de texte sublime

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. Si votre réseau est en ligne, assurez-vous de bien comprendre l’incidence possible des commandes.

Transport et authentification

- Transport : Utilisez HTTPS sur le port `6443` pour l'API AsyncOS, le cas échéant. Utilisez le protocole HTTP/6080 uniquement dans les environnements hors production ou de travaux pratiques.
- Authentification: Cet exemple utilise l'authentification HTTP de base. Indiquez l'en-tête `Autorisation : Base <base64(username:password)>`. Ne collez pas de véritables informations d'identification dans le script.

Informations générales

Pour ce document :

Objectif : Ajoutez un dictionnaire à l'ESA à l'aide d'un script Python et de l'API AsyncOS (API REST).

Résultat escompté : L'API renvoie une réponse positive et le nouveau dictionnaire apparaît dans la configuration ESA.

Configurer

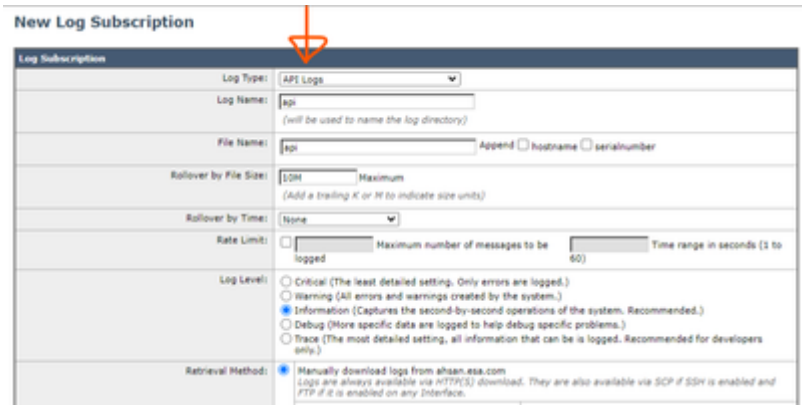
Activez le service API AsyncOS (API REST) sur l'interface de gestion ESA avant d'exécuter le script.

1. Dans l'interface graphique utilisateur, accédez à la page réseau ou d'interface qui contient les paramètres HTTP et HTTPS de l'API AsyncOS.
2. Activez AsyncOS API HTTPS et vérifiez le port 6443.
3. (Facultatif) Activez le protocole HTTP de l'API AsyncOS et vérifiez le port 6080 pour une utilisation en laboratoire.
4. Cliquez sur Save.

Image 1: Vérifiez que le protocole HTTPS de l'API AsyncOS (port 6443) est activé sur l'interface de gestion.

Configurez la journalisation API pour dépanner les requêtes API AsyncOS.

1. Dans l'interface graphique utilisateur, accédez à Administration système > Inscriptions au journal.
2. Cliquez sur Ajouter un abonnement au journal.
3. Dans la liste déroulante Log Type, sélectionnez API Logs.
4. Entrez les autres champs obligatoires, tels que le nom du journal et le nom du fichier, puis cliquez sur Envoyer.



Sélectionnez API Logsdans la liste déroulante Log Type et entrez les autres détails requis.

Remarque : Les journaux peuvent être configurés à partir de l'interface utilisateur graphique.

Script

Utilisez la méthode HTTP `POST` du module de `requêtes` Python pour créer un dictionnaire.

Utilisez les paramètres suivants pour la demande `POST` :

Paramètre	Emplacement	Requis	Remarques
`type_périphérique`	Chaîne de requête URL	Oui	Réglé sur `sea`.
`ignorer la casse`	charge utile JSON	Oui	`0` = sensible à la casse, `1` = non sensible à la casse.
`mots entiers`	charge utile JSON	Oui	`0` = correspondance de sous-chaîne, `1` = correspondance de mot entier.
`codage`	charge utile JSON	Oui	Utilisez `utf-8` sauf si un codage différent est requis.
`paroles`	charge utile JSON	Oui	Tableau d'entrées. Utilisez `["term"]` pour un terme littéral. Pour les identificateurs intelligents, utilisez un format de tuple tel que `["*credit", 2, "prefix"]` comme pris en charge par l'ESA.
`mode`	Chaîne de requête URL	Non	Requis uniquement pour les déploiements en cluster/de groupe (par exemple, `mode=cluster`).

```
import base64
import requests

# ESA management IP address or FQDN
host = "<ESA_IP>"

# Dictionary name to create
dictionary_name = "<DICT_NAME>"

# AsyncOS API endpoint (HTTPS recommended)
url = f"https://{host}:6443/esa/api/v2.0/config/dictionaries/{dictionary_name}?device_type=esa"

payload = {
    "data": {
        "ignorecase": 0,
        "wholewords": 1,
        "words": [
            ["*credit", 2, "prefix"],
            ["*aba"],
            ["Example Term"]
        ],
        "encoding": "utf-8"
    }
}

# Basic authentication header
username = "<USERNAME>"
password = "<PASSWORD>"
```

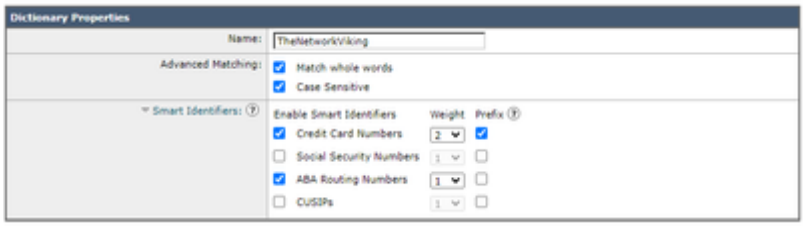
```
creds = base64.b64encode(f"{username}:{password}".encode()).decode()

headers = {
    "Content-Type": "application/json",
    "Authorization": f"Basic {creds}"
}

response = requests.post(url, headers=headers, json=payload, timeout=30, verify=False)
print(f"Status code: {response.status_code}")
try:
    print(response.json())
except ValueError:
    print(response.text)

if response.status_code != 201:
    raise SystemExit("Dictionary creation failed.")
```

Dans le script, le nom du dictionnaire (
) est inclus dans le chemin de l'URL de la demande.

Incluez les paramètres requis (**ignorecase**, **wholewords**, **words**, et **encoding**) dans la charge utile JSON. Ajouter **device_type** à la chaîne de requête d'URL.

Le mode paramètre est requis uniquement lorsque les ESA sont dans une grappe ou un groupe. Dans ce cas, ajoutez `mode=cluster` à la chaîne de requête d'URL de la même manière que device_type.

*credit et*abasont ajoutés à la charge utile pour activer les identificateurs intelligents pour les numéros de carte de crédit et les numéros de routage ABA, comme indiqué dans la capture d'écran :



Le terme Example Term est ajouté à la charge utile en tant qu'élément de dictionnaire correspondant.

Vérifier

Lorsque la requête `POST` réussit, le journal API contient une entrée similaire à celle-ci :

```
API log example
Thu October 5 12:58:19 2023 Info: 198.51.100.10 - - 05/Oct/2023 12:58:19 +0000 POST /esa/api/v2.0/confi
```

Réponse attendue :

```
{"data": {"message": "Added Successfully"}}
```

Si le dictionnaire existe déjà et que la requête `POST` s'exécute à nouveau, le journal de l'API contient une entrée semblable à celle-ci :

```
API log example
Thu October 5 13:00:31 2023 Info: 198.51.100.10 - - 05/Oct/2023 13:00:31 +0000 POST /esa/api/v2.0/confi
```

Réponse attendue :

```
{"error": {"message": "Dictionary already exists.", "code": "409", "explanation": "409 = Request confli
```

Informations connexes

Utilisez les ressources externes suivantes pour obtenir des informations supplémentaires (la procédure décrite dans ce document est autonome) :

Ajout de dictionnaires à l'aide de REST API + Python

Ce document inclut la méthode d'authentification requise et la procédure minimale. Utiliser l'authentification HTTP de base avec l'en-tête `Autorisation : Basic <base64(username:password)>` et privilégiez HTTPS sur le port `6443`.

API REST et Cisco ESA

De plus, le code Python utilisé dans cette procédure peut être extrait de Postman. Si nécessaire, consultez ce didacticiel vidéo :

Postman - Comment générer un script Python

À propos de cette traduction
Cisco a traduit ce document en traduction automatisée vérifiée par une personne dans le cadre d'un service mondial permettant à nos utilisateurs d'obtenir le contenu d'assistance dans leur propre langue.

Il convient cependant de noter que même la meilleure traduction automatisée ne sera pas aussi précise que celle fournie par un traducteur professionnel.